N-gram models for Text Generation in Hindi Language

Tejashree Ghude, Roshni Chauhan, Krushna Dahake, Atharv Bhosale and Tushar Ghorpade*

Department of Computer Engineering
Ramrao Adik Institute of Technology
D Y Patil Deemed to be University

Abstract - Native language plays a vital role for communication. Hindi is preferred by most Indians and it is the fifth most spoken language in the world. Hence, to make User Experience more effective while interacting with Software Applications , we aim to build a Model using Natural Language Processing which takes a specific word as an input and predicts the subsequent words for completing the sentence.It will act as a tab- complete function in Hindi language. This will also pave a way as a use case for building chatbots, personal emails, applications which are content based like cooking recipes, blogs, etc. in Hindi Language.

Keywords : Natural Language Processing, N-Gram, Language Model, Text-Generation.

1 . Introduction

To overcome the language barrier in software applications, a NLP based model will play an effective role to enhance the user experience while interacting with Software Applications. Natural language processing (NLP) is a branch of artificial intelligence that helps computers understand, interpret and manipulate human language [1]. NLP draws from many disciplines, including computer science and computational linguistics, in its pursuit to fill the gap between human communication and computer understanding. NLP combines computational linguistics rule-based modeling of human language with statistical, machine learning, and deep learning models. Together, these technologies enable computers to process human language in the form of text or voice data and to interpret its meaning. This system addresses the problem of predicting the subsequent words, given an initial fragment of text. This approach of autocompleting will prove to be effective for error detection in Hindi language. Instance-based learning and N-gram models can be used to conjecture completions of sentences.

Further the paper is divided as follows: - in the section 2, Literature Survey is discussed, section 3 includes the problem definition of the system, section 4 contains Proposed System, section 5 and section 6 shows the Results and Analysis respectively for the system and conclusion along with future at end of the paper.

2 . Literature Survey

Literature survey provides an idea of the current condition and background information of the topic. In these papers various existing technologies and algorithms are mentioned. Following papers are one of the important papers.

In a research paper by Sharmila Mani et. al.[2], Optimized N-gram is proposed for faster Word Completion and Next Word Prediction (NWP) for mobile phones in real time. N-gram model is trained in various languages like English, Marathi, Hindi, Kannada, Tamil etc and optimised using Stupid Backoff approach.

Mrinalini K et. al. [3] explained a S2ST system for travel expressions which can perform translation between English and Hindi. This is based on the statistical machine translation approach. Bilingual Corpus is used as a training dataset.

Nikita P. Desai et. al.[4] has done a taxonomic survey of resources for Hindi Language NLP systems on all the available research and work done till 2021. Various available NLP resources for Hindi Language like corpora and lexical resources, language scripts, preprocessing such as tokenizer, parser, etc and about other toolkits like inltk, indic nlp, StanfordNLP etc are listed. It provided brief information about research done in Hindi NLP systems and their approaches and resources.

M Mittal et. al.[5], has worked on correction of grammatical errors using a hybrid approach in hindi language. Both statistical and rule based approaches are used to detect grammatical errors in Hindi sentences. Indian language Corpora Initiative (ILCI) corpora is used to detect part of speech using Hidden Markov Model Based technique. Hand written Hindi grammar rules are used to detect the errors and then suggestions are provided for correcting it.

From a brief literature survey, it is discovered that a hybrid approach of text generation using multiple predicting algorithms of text generation using multiple predicting algorithms is the best way to make NLP systems. Preparing, processing and cleaning Hindi datasets and methodologies related to it are discovered.

3 . Problem Statement

Native language plays a vital role for communication. The tab-completion feature for Hindi Language is not effectively developed in existing software applications. Natural Language Processing will prove to be an effective

*Corresponding author: tejashreeghude@gmail.com

approach for predicting the subsequent words which can be used in various applications.

4 . Proposed Methodology

When it comes to text generation, following linguistic grammar is necessary. Every spoken language has its own grammar rules. These grammar rules can be applied in text generation using context free grammar. For English there already exists technologies which can provide these grammatical rules at one place. While for Hindi there aren't many resources which can help us in defining and implementing these grammar rules.

Although defining grammar rules and generating sentences using them is an ideal way, it does not give any assurance that the generated sentence will be meaningful. Here instead of using a rule based approach, a probability based approach was followed for text generation. As one cannot possibly cover all grammar rules for a particular language, a probabilistic approach could be more reliable for text generation. So for a probabilistic approach data was collected, processed and sorted based on the probability of their occurrence before using it for prediction. In this way the accuracy of phrase generation of the text generation model is ensured.

4.1 Architecture/ Framework

The aim is to implement an N-gram model for prediction of text generation . An N-gram model is a type of probabilistic language model for predicting the next item following a similar approach as (n − 1) order Markov model. The Markov model predicts that the state of an entity at a particular position in a sequence depends on the state of one entity at the preceding position. Here instead of depending on the probability of occurrence of the whole sentence this system chooses to consider the last few words. By considering the probability of the whole sentence will restrict the possible patterns for text generation. Considering the probability of word using Markov assumption would reduce the complexity and give the text prediction process more flexibility. It doesn't only make it flexible but also makes it more efficient . The Data Cleansing process is first carried out on the data before passing it data to the model. Data cleaning is rather a very crucial process for this model. Without the cleaning process, the dataset is just a bunch of words that the computer doesn't understand.

4.2 Processing and Prediction :

The initial phase of processing involved data cleaning for the model. As mentioned before, the processing of all the text data was carried out in this phase. First break the paragraph data into list sentences. As the text generation is carried out in Hindi language which is in Devnagri script , all the English latin characters were removed from the dataset during the data cleaning process. Along with these characters there are often some numeric characters which are of no use to us. These types of characters were removed along with all the punctuation signs from each sentence. After data cleaning ,the received data is in the form of a list of sentences . After tokenizing each sentence of the previous list is converted into a list of words which is passed to our model. The model counted the occurrences of each n-gram (i.e. a sequence of n words) in the whole data set. After this process, the probability for each combination is calculated. For calculating the probability , the occurrence count of the whole n-gram taken was divided with the occurrence count of the first (n-1) word in the dataset which gave its value between 0 and 1 . This probability is then used for prediction .This n-gram model checks and adds the next word to the sentences from data based on its probability of occurring with last (n-1) words . For example if input is given as " हम ये " these two words i.e. (n-1) words to the trigram (n-gram) model will predict the third word "नहीं" accordingly which is described in figure no. 4.1.

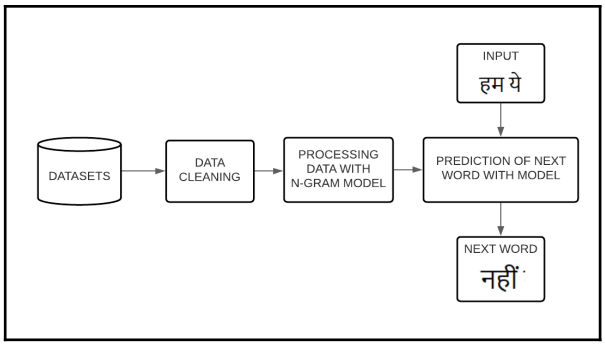

Fig 4.1 Process flow of Model

Text generation could be given as a task of generating text with the goal of appearing indistinguishable to human-written text. This task is also known as "natural language generation". Text generation or natural language generation produce meaningful phrases and sentences in the form of natural language. In essence, it automatically generates text based data in a human-like manner at high speed. Text generation can be addressed with Markov processes or with deep generative models. Here the implemented model is based on the N-gram algorithm. Here different Hindi datasets were used. The data from these dataset is first processed before passing to the model. This model could suggest the next word from the previous word based on the probability of their occurrence.

4.4 Data Preprocessing

The two dataset used here are Hindi Wikipedia Articles - 172k [12], HindiEnglish Corpora (HindEnCorp 0.5 by Charles University) [13]. Hindi Wikipedia Articles - 172k is actually a collection of text files of hindi Wikipedia articles . At first , each article was extracted from the text file and converted to a list of articles ,there were about 1,72,279 articles which were then further broken into a list of sentences to get 26,61,107 sentences. HindiEnglish Corpora (HindEnCorp 0.5 by Charles University) was in csv file form with Hindi and English sentences, Hindi sentences were extracted from this dataset

which were about 97,662 sentences. In total there were **27,88,714** sentences for training our model.

Along with this a small Hindi dataset was created using essays and stories in Hindi to show which was used in the initial stage of model building . This Hindi dataset which was initially in the form of a text file was also cleaned and processed before passing to the model. As mentioned before, the data cleaning process involved removing numbers, special characters and latin characters and at the end a collection of sentences(list of sentences) was generated. These sentences were first tokenized before feeding them to our model .

4.5 N-gram model

An N-gram language model is a technique of counting sequences of characters or words that allows us to support rich pattern discovery in text. For comparison of the accuracy two n-gram models : trigram and bigram were implemented. Bigrams are a set of two consecutive words from the sentences whereas trigram is three consecutive words in a sentence. So first all the possible trigram (i.e. consecutive three word combinations in the corresponding sentence)were extracted.

In the Trigram model the occurrence of each trigram was also counted during extraction. In the model the association of these words is stored based on the occurrence of third with respect to the first word. After getting the count of each pair i.e group of first two words and third word, its count is divided with the count occurrence of first two words. This gives us the probability which was used for prediction. Following figure 4.2 shows how sentences are divided and then the relation in which they are stored .

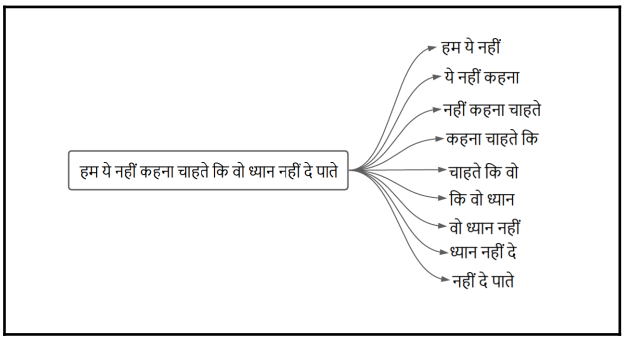

Fig 4.2 Trigram Generation of sentences

After getting the probabilities for each combination (n-gram) in the dataset, text generation was carried out shown in figure 4.3. For text generation ,input of the first two words to the model was given .The model checks all the possible third words and then appends the third word to the sentence if its probability of occurrence is more than or equal to threshold probability i.e 0.1 .The last step is repeated until it reach to condition were the last two word don't have further third word combination, at this condition te process terminates.

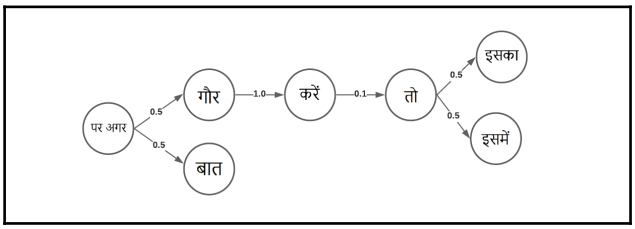

Fig 4.3 Sentence generation

Using these N-grams and the probabilities of the occurrences of certain words in certain sequences could improve the predictions of auto completion systems.

5 . Results

Different inputs were given to the models to analyze their outputs. The trigram takes input of a group of two words while bigram takes only the last word as input. Because of this reason we got different outputs for trigram and bigram .

The input " पर अगर " was given to the Trigram model and it predicted the sentence shown in the next fig 5.1 . The Trigram model stopped after words " पाते हैं " as there would have been no combination with " पाते हैं " as input variable in the model .Whereas when the input " पर " given to the Bigram model it predicted the sentence shown in the fig 5.2 . Similar to the Trigram model, the Bigram model terminates the sentence at " हैं ".

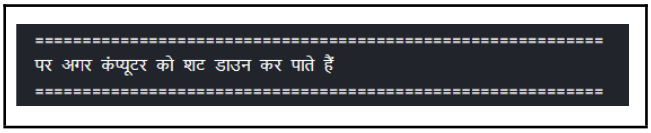

Fig 5.1 Output of Trigram model

Fig 5.2 Output of Bigram model

To show working of the model a small dataset was used to show how the model takes the decision on the basis of probability. After adding each word , the model repeats the last step and checks the possible third word and their probability. The following figure 5.3 is the output of that experiment, where the probability and the third word option model at each step was also printed.

Fig 5.3 Output of with probabilities

The accuracy for this Trigram model was calculated by considering true positive (grammatically correct and present in the dataset) and true negative (grammatically correct and not present in the data set as our data set . Here TP = True Positives, TN = True Negatives, FP = False Positives, and FN = False Negatives. Following is the output we got for the accuracy of the Trigram model which is shown in Figure 5.4.

Fig 5.4 Output of Accuracy of Trigram model

6 . Analysis

The N-gram method gives the assurance of accurate phrase generation. To analyze the model and to find the efficiency of the model, accuracy of the model was checked. The accuracy of our model depends on whether the model is generating grammatically correct sentences or not. The model was generating sentences by predicting the next word for a group of words based on probability of their occurrence in the dataset. There were chances of the model generating a grammatically wrong sentence as only upto Trigram model implemented but nevertheless the accuracy was enough for smooth functioning of text generation model.

To find the accuracy of the model True vs. False and Positive vs. Negative method was used. Here the True vs False condition was set for grammatical correctness of the model. Sentence was assigned True if it is meaningful and grammatically correct, while it was assigned False if it is not meaningful or if it is not grammatically correct. And Positive vs. Negative condition was applied on the basis of whether the sentence is present in our data set or not.

A true positive is an outcome where the model correctly predicts the positive class. Similarly, a true negative is an outcome where the model correctly predicts the negative class. A false positive is an outcome where the model incorrectly predicted the positive class. And a false negative was an outcome where the model incorrectly predicted the negative class.

Accuracy is the fraction of predictions the model got right. Formally, accuracy has the following definition in (1) :

$$Accuracy = \frac{Number\ of\ correct\ predictions}{Total\ number\ of\ predictions} \quad (1)$$

For binary classification, accuracy can also be calculated in terms of positives and negatives as follows in (2) :

$$Accuracy = \frac{TN + TP}{TP + TN + FP + FN} \quad (2)$$

Where TP = True Positives, TN = True Negatives, FP = False Positives, and FN = False Negatives. In this way we have evaluated the accuracy of our model . By calculating the accuracy of sentence generation we have estimated the performance of the model.

In n-gram two models were built. The first is the Bigram model and the second is the Trigram model. The sentence generation in both models is quite accurate but it's more accurate in trigram as the trigram considers the last two words while bigram only considers last words . Usually, n-grams more than 1 is better as it carries more information about the context in general. However the result might differ for different datasets. In bigram one past word is considered and in trigram two past words are considered. It can happen that past two words itself occurred less time and when it happens it contains all those probable words in the same, more or less frequency.This can highly affect the accuracy of the model . So we can say that the accuracy will depend on the training and testing datasets. A testing dataset is created for the training. The testing dataset consisted of pairs of words which we fed to the Bigram and Trigram model to generate sentences. After generating sentences and classifying the result with True vs. False and Positive vs. Negative method as mentioned before the final accuracy we got was 69% for the same Trigram model and 61% of Bigram model.The confusion matrix of the result is given in fig 6.1 .This accuracy test was done on the small dataset. So by increasing the size of the dataset we can surely increase the range of accuracy of our model.

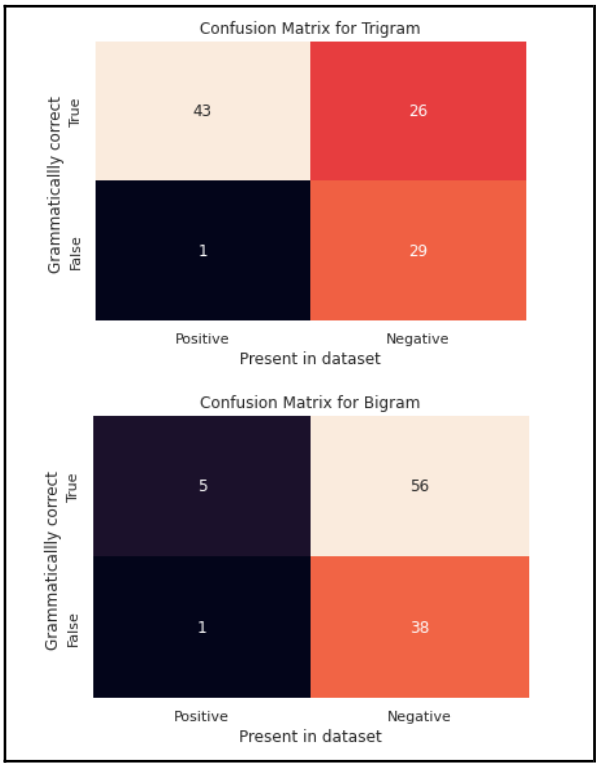

Fig 6.1 Confusion Matrix for Trigram and Bigram

There are many other algorithms which can be used as

support for this model to improve its efficiency and accuracy. Here no other technology was used for grammar structuring for text generation. In spite of using a basic probabilistic approach for the text generation, 69% and 61% accuracy was achieved by the Trigram and Bigram model respectively.

7 . Conclusion

The proposed N-gram approach in this paper ensured the accuracy of the language model. As language is creative and new sentences are generated every minute, It is very difficult to count all possible sentences. But other ways by which this task can be carried out are not as efficient and accurate as probability based prediction. So it is observed that the accuracy of these existing models is not adequate to our current accuracy needs. The N-gram language model could be relatively more accurate .With support of other technologies we can improve its performance in future. Hence, this model can be used in various software applications for tab completions, emails, chatbots, etc.

References

[1] Diksha Khurana, Aditya Koli, Kiran Khatter, Sukhdev Singh, “Natural Language Processing: State of The Art, Current Trends and Challenges”, Researchgate, 2017.

[2] Sharmila Mani, Sourabh Vasant Gothe, Sourav Ghosh, Ajay Kumar Mishra, Prakhar Kulshreshtha, Bhargavi M and Muthu Kumaran, “Real-Time Optimized N-gram For Mobile Devices”, IEEE 13th International Conference on Semantic Computing (ICSC) 2019.

[3] Mrinalini K,Vijayalakshmi P , “Hindi-English Speech-to-Speech Translation System/or Travel Expressions”, ICCPEIC, India 2015.

[4] Nikita P. Desai, Mr.(Dr.) Vipul K. Dabhi, “Taxonomic survey of resources for Hindi Language NLP systems”, Arxiv, 2021.

[5] M Mittal, SK Sharma, A Sethi, “Detection and Correction of Grammatical Errors in Hindi Language”, International Journal of Computer Sciences and Engineering, 2020, 2347-2693.

[6] Ankur Sonawane, Sujeet Kumar Vishwakarma, Bhavana Srivastava, Anil Kumar Singh, “Generating Inflectional Errors for Grammatical Error Correction in Hindi”, Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing, 2019.

[7] Tobias Scheffer, Peter Haider, Steffen Bickel, “Using Hybrid Approach Predicting Sentences using N-Gram Language Models”, International Conference on Advanced Information and Communication Technology, Dhaka, Bangladesh, 2020.

[8] Jayashree Nair, K Amrutha Krishnan, R Deetha, “An Efficient English to Hindi Machine Translation System Using Hybrid Mechanism”, ICACCI, Sept. Jaipur, India 2016.

[9] Tom Young, Devamanyu Hazarika, “Recent Trends in Deep Learning Based Natural Language Processing”,Arxiv, 2018.

[10] B. S. Harish , R. Kasturi Rangan, “A Comprehensive Survey on Indian Regional Language Processing”, Springer, 12 June 2020.

[11] T. Brants, A. C. Popat, P. Xu, F. J. Och, and J. Dean, “Large language models in machine translation,” in Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL), 2007.

[12] https://www.kaggle.com/aiswaryaramachandran/hindienglish-corpora

[13] https://www.kaggle.com/disisbig/hindi-wikipedia-articles-172k

[14] https://www.clarin.eu/resource-families/parallel-corpora